EL SEGMENTO DE TIERRA DEL SATÉLITE UPMSAT-2*

Alejandro Alonso, Raúl Torres, Juan Zamorano, Jorge Garrido, Juan A. de la Puente
Information Processing and Telecommunications Center (IPTC-UPM)
Universidad Politécnica de Madrid
alejandro.alonso@upm.es

Resumen

UPMSat-2 es una misión experimental consiste en el diseño, lanzamiento y operación de un minisatélite en órbita. Su principal objetivo es servir de plataforma de demostración tecnológica para la ejecución de diversos experimentos y para fines didácticos. El grupo STRAST se encarga de desarrollar el software del segmento de tierra y del segmento de vuelo.

La comunicación entre el satélite y el segmento de tierra se lleva a cabo mediante un enlace que permite intercambiar mensajes de telecomandos y de telemetría. El segmento de tierra debe gestionar la conexión con el satélite, intercambiar mensaje y operar con el satélite. Un aspecto importante en este segmento es una interfaz de usuario que permite gestionar los mensajes generados durante toda la vida del satélite.

En este artículo se describen los aspectos más relevantes del desarrollo del segmento de tierra. Los objetivos del desarrollo han sido, obviamente, satisfacer los requisitos identificados y explorar técnicas actuales relacionados con aspectos importantes como disponibilidad o tolerancia de fallos.

La arquitectura de sistema está basada en los requisitos mencionados. Está compuesta de un conjunto de componentes que se pueden ejecutar en un sistema distribuido y compatible su despliegue en la computación en la nube. Un subsistema interacciona con la antena de comunicación para actualizar los datos. El almacenamiento se basa en una base de datos Cassandra, que permite la replicación de los datos para tolerar fallos. La interfaz web está basada en Django y Zeppelin, y su diseño permite disponer de instancias replicadas. El sistema se ha desarrollado en Ada y Python.

Palabras clave: software espacial, segmento de tierra, sistemas de tiempo real

1. Introducción

El proyecto UPMSat-2 tiene por objetivo la construcción de un microsatélite experimental que sirva como demostrador tecnológico y para el despliegue de aplicaciones científicas y educativas. El proyecto está liderado por el Instituto Ignacio da Riva (IDR-UPM), que dirige y coordina la misión y es responsable del diseño y la construcción de la mayor parte de los componentes del satélite, con la colaboración de otros grupos universitarios y empresas del sector espacial. El grupo STRAST (Sistemas de Tiempo Real y Arquitectura de Servicios Informáticos) del IPTC-UPM tiene a su cargo el desarrollo de todo el software de la misión para los segmentos de vuelo y tierra.

UPMSat-2 es un microsatélite con 50 kg de masa, cuya envolvente geométrica es un paralelepípedo de $0{,}5 \times 0{,}5 \times 0{,}6$ m. Su lanzamiento está previsto en septiembre de 2019, sobre una órbita polar de unos 600 km de altitud, con una vida útil estimada de 2 años.

El sistema de gestión de datos (OBDH) está basado en un computador embarcado (OBC) desarrollado por TECNOBIT, con la colaboración del grupo STRAST y del IDR. El computador tiene un procesador LEON3 con arquitectura SPARCv8, 4 MB de memoria RAM, 2 MB de EEPROM, 64 canales de entrada analógica y 112 puntos de E/S digital. Este computador realiza todas las funciones de gestión de datos, control de potencia y control de actitud en el satélite. La carga útil consiste en una serie de experimentos sobre algunos subsistemas, sobre los que se espera poder obtener información sobre su comportamiento en vuelo que, en su caso, se pueda utilizar para su calificación.

El software de la misión UPMSat-2 tiene dos componentes principales:

- Segmento de vuelo: incluye el software embarcado en el satélite, cuyas principales funciones son:
  - Arranque, reinicio y apagado ordenado.
  - Control de modos de funcionamiento e incidencias.

Este artículo ha sido financiado parcialmente por el Plan Nacional de I+D+i, proyecto PRECON-I4 (TIN2017-86520-C3-2-R).

- Control de actitud.
- Recogida de datos de estado interno y navegación.
- Vigilancia del comportamiento de los subsistemas y del estado y configuración.
- Control de comunicaciones con tierra (telemetría y telecomandos).
- Arranque y control de los experimentos.

- Segmento de tierra: incluye el software de la estación de seguimiento en tierra y de otros sistemas complementarios. Sus funciones más importantes son:
  - Determinación de la posición del satélite.
  - Cálculo de parámetros de órbita y tiempos de observación.
  - Recepción, descodificación y procesamiento de mensajes de telemetría.
  - Composición, codificación y transmisión de telecomandos.
  - Gestión de las interfaces del operador y los investigadores
  - Generación de informes y explotación de los datos disponibles.

En este documento se explican los elementos más importantes del desarrollo del segmento de tierra: requisitos fundamentales, arquitectura del software, soluciones técnicas empleadas y entorno de pruebas.

2. El segmento de tierra de la misión UPMSat-2

2.1. Descripción

La figura 1 ilustra la interacción entre el satélite y la estación de la tierra. El satélite tiene una órbita polar heliosíncrona con un período de aproximadamente 97 minutos. Cada 24 horas hay dos períodos de visibilidad del satélite desde la estación de tierra, cada uno de ellos de un máximo de 10 minutos de duración. Durante los períodos de visibilidad, las comunicaciones con la nave espacial se llevan a cabo mediante un enlace de radio dual en la banda VHF de 400 MHz, con una tasa de transferencia de 9600 bit/s. Durante el resto de la órbita se emiten periódicamente mensajes de telemetría básica en una frecuencia de aficionados en la misma banda de VHF.

En un periodo de visibilidad se intercambian dos tipos de mensajes:

- Telecomandos (TC), enviados desde tierra al satélite. Estos mensajes se usan para controlar el comportamiento del satélite, su modo de funcionamiento, la configuración de los sensores, la configuración del algoritmo de control de actitud, y el inicio y fin de los experimentos.

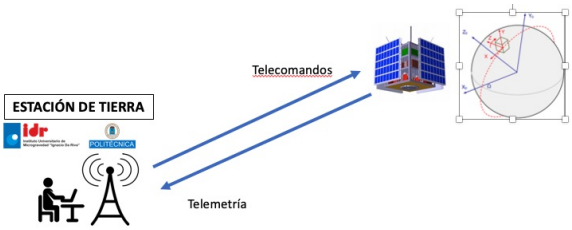

Figura 1: Comunicación entre el satélite y el segmento de tierra.

- Telemetría (TM), mensajes enviados desde satélite a la estación de tierra. Los más relevantes son:
  - *Hello*: Información sobre el estado actual del sistema.
  - *Event*: Información de sucesos o errores relevantes.
  - *Housekeeping*: Datos completos de todos los sensores del sistema.
  - *Experiment*: Datos de los sensores activos en el experimento.

2.2. Requisitos básicos

Los requisitos básicos funcionales se han descrito en la sección 1. Además, es preciso cumplir los siguientes requisitos no funcionales:

- Disponibilidad: Todos los mensajes de telemetría recibidos deben almacenarse y estar disponibles para los usuarios (operadores e investigadores) en todo momento. El objetivo es que se pueda consultar siempre la información intercambiada entre el satélite y la estación de tierra, puesto que futuras emisiones de telecomandos pueden depender de alguna telemetría recibida. Para que los datos estén disponibles en todo momento se deben habilitar los mecanismos de replicación necesarios para garantizar su integridad.
- Tolerancia a fallos y fiabilidad: el sistema debe ser capaz de responder ante fallos, tanto en emisión como en recepción. De esta manera, es imprescindible controlar la replicación de los datos del sistema, realizar copias de seguridad periódicamente y mantener la información en varios equipos separados geográficamente
- Flexibilidad: el desarrollo de la estación de tierra del satélite y la interfaz de usuario con-

lleva mucho esfuerzo de programación. Es primordial que se garantice las flexibilidad del software, de modo que sea sencillo de modificar para adaptarlo a los cambios que requiera el comportamiento del satélite.

- Seguridad: Sólo los usuarios con permiso deben poder acceder y modificar los datos almacenados. Un acceso no autorizado a los datos de telemetría o un envío malicioso de telecomandos podría resultar en un ataque que inutilizara una misión de enorme envergadura y valor.

3. Arquitectura de software

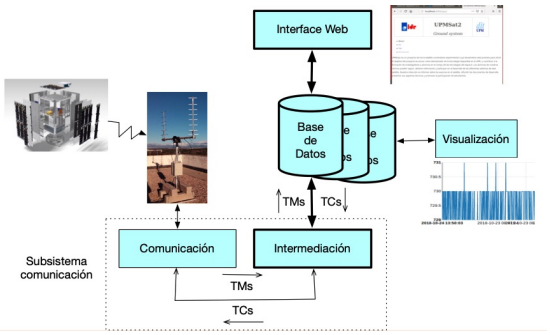

Figura 2: Arquitectura de software del segmento de tierra.

En la figura 2 se muestran los componentes del sistema y sus conexiones. A continuación, se proporciona una breve descripción de estos componentes:

- Interfaz web: es el componente desde el que el usuario puede crear telecomandos para enviar al satélite y consultar la telemetría recibida. Todos los datos que utiliza los extrae directamente a partir de consultas a la base de datos sobre la que se apoya.
- Base de datos: módulo encargado de almacenar toda la información que maneja el proyecto UPMSat-2. Se usan varias tablas para almacenar los telecomandos pendientes de enviar, los telecomandos enviados, la telemetría y los parámetros de configuración.
- Subsistema de comunicación: formado por el módulo de comunicación y el módulo de intermediación, es el encargado de transmitir y procesar la información intercambiada entre la interfaz de usuario y el sistema de antena. La información transmitida por el satélite se obtiene de la antena almacenada en fichero, y el subsistema de intermediación realiza el procesamiento necesario para almacenar esta información en la base datos. Por otra parte, el subsistema toma de la base de datos los telecomandos generados desde la interfaz de usuario, y los procesa para producir los ficheros adecuados para su transmisión por la antena.
- Subsistema de antena: es el encargado de intercambiar de forma directa los mensajes de TM y TC con el satélite.
- Subsistema de visualización: módulo adicional que extiende la interfaz de usuario para mostrar los datos intercambiados con el satélite de una forma gráfica, consiguiendo así una lectura más sencilla e intuitiva del estado del satélite.

El diagrama de secuencia en la figura 3 ilustra el comportamiento dinámico de la gestión de telecomandos en la arquitectura. El usuario interacciona con el sistema usando la interfaz web (parte izquierda de la figura). En este proceso, puede crear, examinar, modificar o borrar los TC pendientes de enviar. Estas operaciones se actualizan en la base de datos. El componente de comunicación solicita recibir los TC pendientes en la base de datos al de intermediación, cuando se aproxima la comunicación con el satélite[1]. A continuación, el componente de intermediación transfiere los contenidos de la tabla de TC pendientes a la tabla de TC enviados. A partir de este momento, los contenidos de esta tabla no se pueden modificar. El módulo de comunicación espera a recibir un mensaje del satélite para indicar que hay visibilidad. Cuando esto ocurre se envían todos los TC al satélite. El módulo de comunicación puede recibir un mensaje de reconocimiento del subsistema de la antena, indicando que el TC se ha enviado. El satélite, por su parte, no confirma inmediatamente la recepción del TC, que se comprueba únicamente al recibir el mensaje de telemetría de respuesta.

4. Decisiones técnicas

4.1. Base de datos

La base de datos es un componente fundamental de software de tierra, ya que debe mantener toda la historia de TC y TM para permitir en todo momento el análisis del comportamiento del satélite y de los experimentos realizados. En el análisis, se definieron dos tipos de requisitos. En primer lugar, es necesario un rendimiento adecuado de la base de datos. Aunque en en este caso no se esperan grandes problemas en cuanto al rendimiento del sistema, el diseño se ha efectuado teniendo en cuenta la posible extensión del sistema a otras

[1]La dinámica del satélite permitirá conocer, con bastante precisión, cuándo habrá cobertura

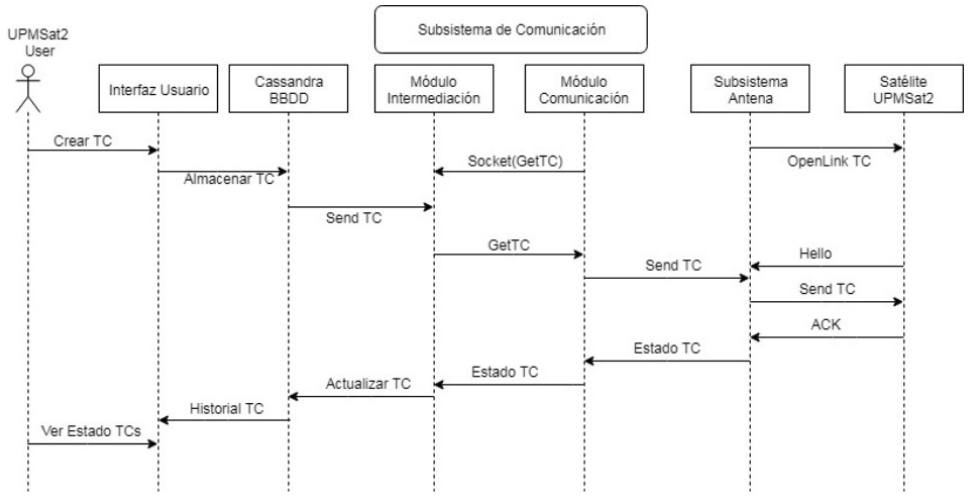

Figura 3: Diagrama de secuencia de envío de Telecomandos.

aplicaciones en las que se realicen tratamientos de datos masivos en entornos similares. El segundo tipo de requisito es la tolerancia de fallos. El sistema debe mantener la información aunque falle un dispositivo básico, ya que los datos obtenidos del satélite no se pueden reemplazar en ningún caso.

La exploración de las herramientas disponibles aconsejó el uso del software de base de datos *Apache Cassandra* [7]. Este software está orientado a gestionar sistemas temporales, y es capaz de gestionar datos masivos, con un tiempo de acceso muy eficiente, con bibliotecas para integrar con varios lenguajes de programación, y redundancia de la base datos muy fácil de configurar. *Cassandra* es un sistema no SQL, por lo que no es una solución adecuada para todos los tipos de sistemas, pero en este caso resulta muy interesante por su capacidad para tratar series temporales asociadas a los mensajes.

4.2. Lenguajes de programación

El lenguaje Ada [10] ha sido una herramienta fundamental en el desarrollo del sistema de software vuelo, y en la conexión con la antena y gestión de los mensajes en la estación de tierra. Sin embargo, al realizar un prototipo de la interacción entre estos componentes con la base de datos los resultados no fueron totalmente satisfactorios. Aunque la integración con la base de datos es posible, la biblioteca de acceso a la misma es mejorable, y falta flexibilidad en los cambios de la configuración de la base de datos. En consecuencia, Ada se utilizó únicamente en el componente de comunicación.

Como alternativa se utilizó *Python* [8], un lenguaje interpretado que soporta varios paradigmas: programación orientada a objetos, programación imperativa y programación funcional. Es un lenguaje muy maduro, muy utilizado y con un gran número de bibliotecas disponibles. Sus características permiten desarrollar código eficiente y flexible de mantener. Para el proyecto se ha utilizado una biblioteca que se integra perfectamente con *Cassandra*. Por estas razones, se decidió usar este lenguaje en el componente de intermediación para mantener y gestionar los mensajes en la base de datos.

La biblioteca de integración con la base de datos se basa en el modelo ORM (Object-Relational Mapping). Este paradigma relaciona las tablas de la base de datos con clases del lenguaje de programación. Por ejemplo, la forma de generar una fila de una tabla requiere crear un objeto equivalente, rellenar los datos del mismo y guardarlo directamente en la base de datos, sin que sea necesario conocer los clásicos lenguajes de gestión de las bases de datos, como SQL.

Para conectar los componentes de comunicación e intermediación, al estar implementados en dos lenguajes de programación diferentes, se consideraron dos alternativas diferentes. La primera consiste en usar una biblioteca de Ada para conectar con Python. La segunda opción es utilizar una conexión con *sockets* entre ambos componentes, utilizando un lenguaje intermedio para enviar y recibir los mensajes. Esta es la solución adoptada, ya que permite desacoplar la gestión de la antena y el mantenimiento de los datos y la interfaz de la web. Los usuarios pueden interaccionar con los mensajes de TC y TM almacenados en la base de datos sin tener que usar en el computador conectado a la antena.

El lenguaje intermedio utilizado para la interac-

ción entre los dos componentes se basa en el lenguaje XML. Se han definido dos esquemas de TM y TC. El componente de comunicación realiza la transformación entre los ficheros de XML y los mensajes enviados o recibidos. El componente de intermediación realiza las operaciones equivalentes entre los ficheros recibidos y el código de acceso a la base de datos.

4.3. Interfaz web

Hay una gran variedad de herramientas *(frameworks)* para generar interfaces web e integrarlas con otras herramientas, como bases de datos o software de visualización. En este caso se consideró como primera opción el uso de una herramienta programada en Python, con objeto de no utilizar lenguajes de programación adicionales. La herramienta seleccionada es DJango [2], un *framework* maduro, basado en paradigmas actuales, como MVC (Model View Controller), y con bibliotecas que facilitan la integración con *Cassandra*. La experiencia con DJango ha sido muy satisfactoria.

4.4. Herramientas de visualización

El tiempo de vida previsto para la misión UPMSat-2 es de dos años. El software embarcado en el satélite toma medidas de todos los sensores cada minuto, y enviar los valores correspondientes a tierra. También hay que enviar a tierra los valores de los sensores de los experimentos cuando estén activos. Todo ello resulta en una enorme cantidad de datos recibidos en la estación de tierra, que se deben analizar para estudiar el comportamiento del satélite y los resultados de los experimentos. Para facilitar esta tarea se se han explorado distintas opciones de software de visualización de datos, entre las que se ha seleccionado para facilitar este esfuerzo. Finalmente, se seleccionó la herramienta de visualización de datos *Zeppelin* [1], que está bien integrada con *Cassandra* y dispone de numerosos intérpretes de *backend* para la programación y generación de informes y gráficas.

Actualmente se está empezando a usar esta herramienta. Se han desarrollado dos tipos de informes que se generan automáticamente cuando se conecta con el satélite y cuando se ejecuta un experimento, respectivamente. Con su ayuda, los operadores e investigadores pueden acceder a la información recibida de forma completa e inmediata.

5. Pruebas del segmento de tierra

En el desarrollo del software de tierra se ha seguido un ciclo de vida de pruebas en V tradicional. En primer lugar se han efectuado pruebas unitarias para comprobar el correcto funcionamiento de cada componente del sistema. Más adelanta se han llevado a cabo pruebas de integración al ir conectando los componentes de la arquitectura. En este proceso se han utilizado suplentes (*stubs*) para emular el comportamiento del satélite.

En la fase final se han realizado pruebas del sistema completo. Para ello, se ha usado la maqueta que se muestra parcialmente en la figura 4. Se trata de disponer de un sistema equivalente al mostrado en la figura 1. En la fotografía se puede observar el satélite completo, ya dispuesto para el lanzamiento. El satélite está conectado mediante una línea serie a un computador que interacciona con un monitor de software instalado en el computador embarcado. De esta manera se pueden ejecutar órdenes para cargar el software embarcado en el satélite, arrancarlo y recibir trazas de la ejecución. El satélite está conectado con una antena, que se conecta a otra antena equivalente situada en una sala, y que está conectada al computador que ejecuta el software de comunicación en tierra. El satélite y el computador de comunicaciones, por tanto, intercambian mensajes de TM y TC reales mediante las antenas. La interfaz web permite generar telecomandos y acceder a los mensajes de telemetría recibidos. De esta forma es posible validar el comportamiento completo del software de la misión en condiciones cercanas a las reales.

Las pruebas realizadas hasta ahora han sido satisfactorias. Actualmente, se están realizando pruebas adicionales, previas al sellado del satélite para su transporte a la base de lanzamiento.

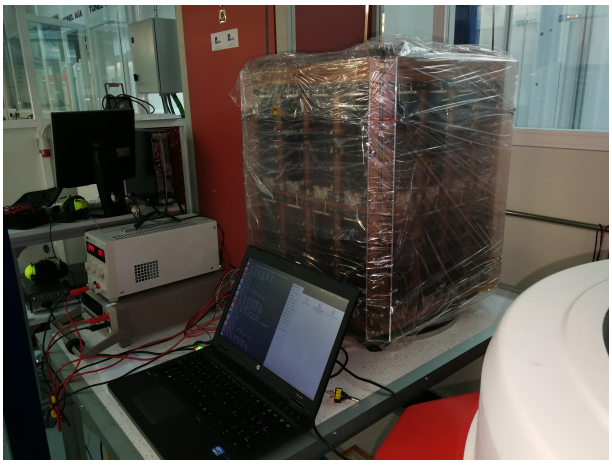

Figura 4: Ilustración del sistema de prueba del satélite.

6. Conclusiones

Este artículo describe aspectos relevantes del desarrollo del segmento de tierra de la misión UPMSat-2. En este desarrollo, se han perseguido dos propósitos principales: satisfacer los requisitos del proyecto y emplear técnicas actuales. Los requisitos funcionales básicos han sido: gestionar la comunicación con el satélite, mantener los mensajes enviados en la comunicación y proporcionar una interfaz para que los operadores e investigadores puedan controlar el comportamiento del satélite y analizar toda la información relacionada con el vuelo del satélite.

Las funciones generales de este segmento consisten en realizar las comunicaciones con el satélite y gestionar los mensajes intercambiados. La arquitectura de software del sistema ha servido para dirigir la implementación del sistema. En la arquitectura, destacan los siguientes aspectos:

- Integración satisfactoria entre el componente de comunicación con la antena y el resto del sistema.
- Desacoplamiento de este componente con los componentes encargados de gestionar la base de datos.
- Redundancia de la base de datos y la interfaz web

Un aspecto importante del desarrollo ha sido explorar técnicas actuales para satisfacer los requisitos funcionales y, en especial, los no funcionales. Se ha usado la base de datos *Cassandra* por su eficiencia de acceso y su facilidad de configuración y despliegue de la redundancia de los datos. El lenguaje Ada es ideal para integrarse con la antena y asegurar un comportamiento predecible en el envío y recepción de mensajes.

En relación al lenguaje Python, es destacable la eficiencia en el proceso de desarrollo, la disponibilidad de bibliotecas para integrar el código con otras herramientas usadas, y la flexibilidad en su evolución y mantenimiento. El uso de la herramienta de aplicaciones web *Django* ha sido muy satisfactorio. Es una herramienta muy madura e integrada con otros aspectos técnicos en este desarrollo.

El prototipo del segmento de tierra se ha probado en una maqueta, integrada con el satélite completo, en la que se implementa la comunicación entre ambos sistemas mediante el sistema de radio real de la misión. Las pruebas han sido satisfactorias y permiten confiar en que el comportamiento del satélite en vuelo será correcto. En esta fase, se han experimentado todos los mensajes de telemetría y telecomando, y se ha evaluado la interfaz web y la base de datos. Actualmente se están refinando algunos elementos del segmento de tierra, y se está desarrollando el software de informes y acceso a datos de visualización.

English summary

THE GROUND SEGMENT OF THE UPMSAT-2 SATELLITE

Abstract

UPMSat-2 is an experimental mission aimed at designing, launching and operating a mini-satellite on orbit. Its main objetive is to develop a technological demonstration platform that can be used to carry out several experiments and as a teaching aid. The STRAST group is in charge of developing de software system for both the flight and ground segments,

Communication between the flight and ground segments is performed by means of a radio link that enables telecommand and telemetry messages to be exchanged. The ground segment must manage the communications with the satellite, exchange the required messages, and operate the satellite based on them. An important feature of the ground segment is to develop a user interface that allows operators to manage all the messages exchanged during the lifetime of the satellite.

This article is aimed at describing the most important aspects of the development of the ground segment software. The main objetives of this development are complying with the identified requirements and exploring the use of state-of-the-art techniques related to important characteristics, such as availability and fault tolerance.

The system architecture is based on these requirements. It consists of a set of components that can run on a distributed platform, and are compatible with deployment on a cloud environment. There is a devoted subsystem for interacting with communication antenna for data exchange. Data storage is based on a Cassandra database,

which provides for data replication for fault tolerance. There is a web-oriented user interface based on Django and Zeppelin, designe din a way allowing replicated instances. The system has been developed using Ada and Python as programming languages.

Keywords: Space software, ground segment, real-time systems.

Referencias

[1] Apache Zeppelin. Web-based notebook. https://zeppelin.apache.org/, 2018.

[2] Django web framework. https://www.djangoproject.com, 2018.

[3] Alejandro Alonso, Emilio Salazar, and Juan A. de la Puente. Design of on-board software for an experimental satellite. In *Jornadas de Tiempo Real — JTR-2013*, 2103.

[4] Juan A. de la Puente, Jorge Garrido, Emilio Salazar, Juan Zamorano, and Alejandro Alonso. Using internet-based technologies in a university satellite project. *IFAC-PapersOnLine*, 48(29):82 – 86, 2015.

[5] Juan A. de la Puente, Juan Zamorano, Alejandro Alonso, Jorge Garrido, Emilio Salazar, and Miguel A. de Miguel. Experience in spacecraft on-board software development. *Ada User Journal*, 35(1):55–60, March 2014.

[6] Jorge Garrido, Juan Zamorano, Juan A. de la Puente, Alejandro Alonso, and Emilio Salazar. Ada, the programming language of choice for the UPMSat-2 satellite. In *Data Systems in Aerospace — DASIA 2015*. Eurospace, 2015.

[7] Eben Hewitt and Jeff Carpenter. *Cassandra: The Definitive Guide, 2nd Edition*. O'Reilly Media, 2016.

[8] Mark Lutz. *Learning Python, 5th Edition*. O'Reilly Media, 2013.

[9] Santiago Pindado, Elena Roibás-Millán, Javier Cubas, Andrés García, Angel Sanz, Sebastián Franchini, María Isabel Pérez-Grande, Gustavo Alonso, Javier Pérez-Álvarez, Felix Sorribes, Antonio Fernandez-López, Mikel Ogueta-Gutierrez, Ignacio Torralbo, Juan Zamorano, Juan Antonio de la Puente, Alejandro Alonso, and Jorge Garrido. The UPMSat-2 satellite: an academic project within aerospace engineering education. In *2nd Annual International Conference on Engineering Education & Teaching*, 2017.

[10] S. T. Taft, R. A. Duff, R. L. Brukardt, E. Plöedereder, and P. Leroy, editors. *Ada 2005 Reference Manual. Language and Standard Libraries. International Standard ISO/IEC 8652:1995/Amd 1:2007*. Number 4348 in Lecture Notes in Computer Science. Springer-Verlag, 2006.